C interfaces to GALAHAD SCU

Jari Fowkes and Nick Gould
STFC Rutherford Appleton Laboratory
Thu Jun 22 2023

Chapter 1

GALAHAD C package scu

1.1 Introduction

1.1.1 Purpose

Compute the the **solution to an extended system of** $n+m$ **sparse real linear equations in** $n+m$ **unknowns,**

$$(1)\quad \begin{pmatrix} A & B \\ C & D \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$$

in the case where the n by n matrix A is nonsingular and solutions to the systems

$$Ax = b \text{ and } A^T y = c$$

may be obtained from an external source, such as an existing factorization. The subroutine uses reverse communication to obtain the solution to such smaller systems. The method makes use of the Schur complement matrix

$$S = D - CA^{-1}B.$$

The Schur complement is stored and factorized as a dense matrix and the subroutine is thus appropriate only if there is sufficient storage for this matrix. Special advantage is taken of symmetry and definiteness in the coefficient matrices. Provision is made for introducing additional rows and columns to, and removing existing rows and columns from, the extended matrix.

Currently, only the control and inform parameters are exposed; these are provided and used by other GALAHAD packages with C interfaces.

1.1.2 Authors

N. I. M. Gould, STFC-Rutherford Appleton Laboratory, England.

C interface, additionally J. Fowkes, STFC-Rutherford Appleton Laboratory.

Julia interface, additionally A. Montoison and D. Orban, Polytechnique Montréal.

1.1.3 Originally released

March 2005, C interface January 2022.

1.1.4 Method

The subroutine galahad_factorize forms the Schur complement $S = D - CA^{-1}B$ of A in the extended matrix by repeated reverse communication to obtain the columns of $A^{-1}B$. The Schur complement or its negative is then factorized into its QR or, if possible, Cholesky factors.

The subroutine galahad_solve solves the extended system using the following well-known scheme:

1. Compute the solution to $Au = b_1$;
2. Compute x_2 from $Sx_2 = b_2 - Cu$;
3. Compute the solution to $Av = Bx_2$; and
4. Compute $x_1 = u - v$.

The subroutines galahad_append and galahad_delete compute the factorization of the Schur complement after a row and column have been appended to, and removed from, the extended matrix, respectively. The existing factorization is updated to obtain the new one; this is normally more efficient than forming the factorization from scratch.

1.1.5 Call order

To solve a given problem, functions from the scu package must be called in the following order:

- scu_initialize - provide default control parameters and set up initial data structures
- scu_read_specfile (optional) - override control values by reading replacement values from a file
- scu_form_and_factorize - form and factorize the Schur-complement matrix S
- scu_solve_system - solve the block system (1)
- scu_add_rows_and_cols (optional) - update the factors of the Schur-complement matrix when rows and columns are added to (1).
- scu_delete_rows_and_cols (optional) - update the factors of the Schur-complement matrix when rows and columns are removed from (1).
- scu_information (optional) - recover information about the solution and solution process
- scu_terminate - deallocate data structures

See Section **??** for examples of use.

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

Chapter 3

File Documentation

3.1 galahad_scu.h File Reference

```
#include <stdbool.h>
#include <stdint.h>
#include "galahad_precision.h"
#include "galahad_cfunctions.h"
```

Data Structures

- struct scu_control_type
- struct scu_inform_type

Functions

- void scu_information (void **data, struct scu_inform_type *inform, int *status)
- void scu_terminate (void **data, struct scu_control_type *control, struct scu_inform_type *inform)

3.1.1 Data Structure Documentation

3.1.1.1 struct scu_control_type

control derived type as a C struct

Data Fields

bool	f_indexing	use C or Fortran sparse matrix indexing

3.1.1.2 struct scu_inform_type

inform derived type as a C struct

Data Fields

int	status	return status. A non-zero value indicates an error or a request for further information. See SCU_solve for details.
int	alloc_status	the return status from the last attempted internal workspace array allocation or deallocation. A non-zero value indicates that the allocation or deallocation was unsuccessful, and corresponds to the fortran STAT= value on the user's system.
int	inertia[3]	the inertia of S when the extended matrix is symmetric. Specifically, inertia(i), i=0,1,2 give the number of positive, negative and zero eigenvalues of S respectively.

3.1.2 Function Documentation

3.1.2.1 scu_information()

```
void scu_information (
            void ** data,
            struct scu_inform_type * inform,
            int * status )
```

Provides output information

Parameters

in,out	*data*	holds private internal data
out	*inform*	is a struct containing output information (see scu_inform_type)
out	*status*	is a scalar variable of type int, that gives the exit status from the package. Possible values are (currently): • 0. The values were recorded succesfully

3.1.2.2 scu_terminate()

```
void scu_terminate (
            void ** data,
            struct scu_control_type * control,
            struct scu_inform_type * inform )
```

Deallocate all internal private storage

Parameters

in,out	*data*	holds private internal data
out	*control*	is a struct containing control information (see scu_control_type)
out	*inform*	is a struct containing output information (see scu_inform_type)